L4微内核技术浅析

陈 斐

(江南计算技术研究所,江苏无锡 214083)

摘 要:微内核很小但是很灵活。传统的和非传统的操作系统都可以建立在微内核之上,也适合在微内核上运行。L4是一种可以在alpha上实现的微内核,它可以用C和汇编语言来实现。特别的,L4支持可扩展性和定制性,支持包含可靠性和容错处理的健壮性,保护和安全性,并支持Linux API和ABI,本文对L4做简要概述,介绍了L4的主要概念、特性和技术。

关键词:L4;操作系统;微内核

Analyzing of L4 Microkernel Technique

CHEN Fei

(Jiangnan Institute of Computing Technology, Wuxi 214083, China)

Abstract: Microkernels are minimal but highly flexible kernels. Both conventional and unconventional operating systems can be built on top of or adapted to run on top of them. L4 is the microkernel which can run on alpha and implemented with C or assembly language. It should particularly support extensibility and customizability, robustness including reliability and fault tolerance, protection and security, and it support Linux API and ABI. In this paper we present a brief overview of L4, describe the main concepts, features and technologies of L4.

Key words: L4, operating system, microkernel

1 引言

进入21世纪,计算机技术日新月异,操作系统项目的研究,开始朝着研制能够有效管理和使用,以及适应对性能、安全、容错、升级需求的方向发展。主要的研究目标包括:易于使用,支持体系结构的创新,动态的管理策略,支持故障处理,可扩展,自感知和优化等等。

而现有操作系统一般采用集中式关键代码路径、全局数据结构和全局策略,所采用的技术已经相对落后。为了获得一个具有高可维护性和高可扩展性的系统,我们又把眼光重新投向微内核技术上。

L4是第二代微内核接口。一开始L4是用汇编语言编写的,随后,L4开始由高级语言,如C和C++编写,它可以接口到不同的体系结构(例如Alpha,ARM,MIPS)。

L4第二代微内核的体系结构如图1所示,它的设计思想主要基于以下概念:内核只提供地址空间管理,内存分配,线程创建,销毁,消息传递等基本的操作功能,其他所有的服务,如文件操作,网络协议处理,甚至设备驱动等都交给用户态应用程序来处理。每个服务有它自己的地址空间。以此,增大了系统的灵活性。同时,内核可以针对特定的平台做深度的优化。

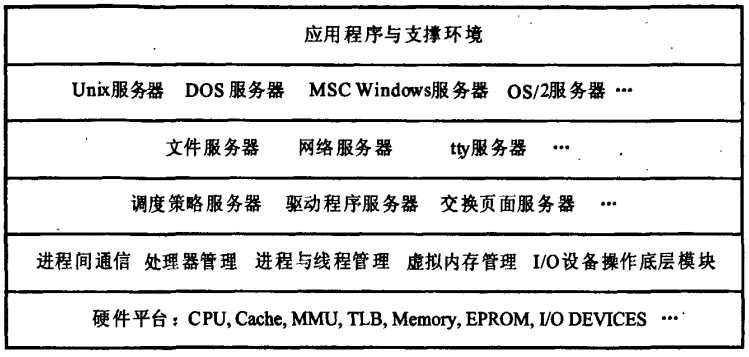

图 1 微内核体系结构

2 以 L4 为代表的第二代微内核的产生和特点

总的来说，微内核设计带来了良好性能。但是，它的设计有一个重要缺点，那就是消息传递的步骤过多，用户态和核心态之间的切换频繁，以及不同地址空间之间的切换过多，这些都导致了以 mach 为主要代表的第一代微内核性能比它设计初期的预想要低得多。也是 mach 最终没有流行起来的重要原因。

解决微内核设计性能问题的一个方法是扩大微内核并把一些关键的服务程序和驱动程序重新加入到内核中去，从而减少系统在用户态和核心态之间的切换，以及系统在不同地址空间之间的切换。这方面的例子有 Windows NT 4.0 的设计，这个版本的 Windows NT 把本来运行在用户态的图形系统又重新加入到内核中，大大提高了图形系统的性能。

但是，扩大内核的方法大大削弱了微内核思想带来的优点，扩大的内核降低了系统的扩展性、灵活性和可靠性。与扩大内核的思路相反，解决微内核性能问题的另一条思路是进一步减少内核的大小并对远程过程调用进行直接优化。这种思路导致了被称为第二代微内核的一些新的内核设计的出现。在这些新的微内核中，L4 微内核是一个著名的例子。

L4 微内核只将基本的操作功能留在内核中，从而进一步缩小内核的设计，带来的一个重要优点是大大提高了操作系统的兼容性，使得基于微内核的操作系统能够模拟其他操作系统的特性。

在 L4 之上的各种服务设施，可以根据平台和需求而定，这样带来的另外一个重要优点是提高了操作系统的可扩展性。微内核设计的一个目标就是定义一个非常好的界面集，从而可以保证系统有条不紊地增加新的功能而得以不断改进。如果在具有主要功能的情况下，系统功能可根据需求而增删，那么这种产品必然会引起更多用户的兴趣。

由于 L4 对操作系统作了进一步抽象，基于微内核的操作系统更容易去掉一些不必要的特性从而被剪裁成一个较小的系统。也就是说，微内核设计使得操作系统有较好的灵活性。

在基于微内核的操作系统上，所有的处理器的相关代码都被封装在微内核中，从而使得操作系统有较好的可移植性。可移植性是指能以最小的重新编码量，在拥有着不同处理器的机器之间进行操作系统的移植的特性。由于与各种不同处理器相关的具体代码与微内核是互相隔离的，因此，在基于微内核的操作系统中，只需作少量改变，就可以把系统从一种处理器移植到另外一种处理器上去运行。

微内核系统的可靠性也较好，在可靠性上，像 Linux 这样的系统都有几百万行代码，并且要经过几年以后才趋于成熟。因此，没有一个程序员能保证使用若干系统服务的应用程序界面代码的正确性，甚至无人敢担保操作系统本身的正确性。而像 L4 微内核这样体积较小的内核可以得到

更多的测试。同时,一些属于传统操作系统内核部分的功能,在 L4 中是由服务程序实现的,所以一旦发生故障不至于导致系统的崩溃。

最后,由于微内核设计基于消息传递机制,所以能更容易支持网络通信。

3 L4 微内核发展现状

L4KA::Pistachio 是已经实现的可以运行在 alpha 体系结构上的 L4 微内核接口。它的编码是用 C++编写的。新发布的 L4KA::Pistachio 增加了 alpha 和其他体系结构的支持。L4KA::Pistachio 建立在以往 7 年的微内核和多服务器研究的基础上,有良好的性能,并支持可移植性。

当前,L4 的 alpha 处理器版本已经在 GNU GPL 下以源码的形式发布,它的端口可以支持全部 64 位 alpha 处理器,包括 21064,21164 和 21264。

在多处理器方面,也已经可以提供完整的,可扩展的和低开销的 SMP 系统支持。并在 21064,21164 和双 CPU 的 21264 系统通过了测试。

4 L4 微内核技术实现

4.1 地址空间

L4 可以做到完全用户级的虚拟内存管理。最初,由根地址空间 σ0 来映像所有的物理内存,可以通过将一个地址空间虚拟内存域映射到下一个地址空间的可访问的虚拟内存域来构建新的地址空间,如图 2 所示(图中的 IPC 是指 Inter—Process Communication,在下文中会有详细叙述)。为了在 σ0 上构建和维护地址空间,微内核提供了三种操作:授权(grant),映射(map)和刷新(flush)。

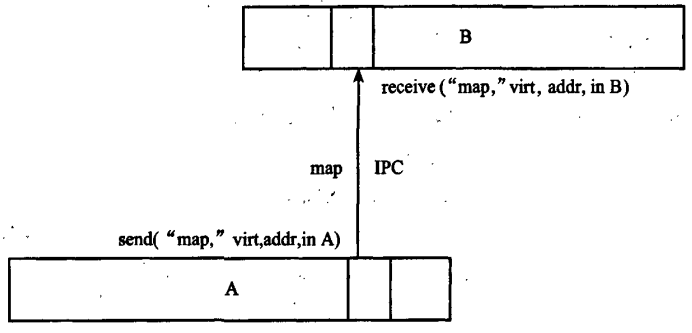

图 2 递归结构地址空间

(1) Grant:倘若接收者同意接收,地址空间的拥有者(也就是授权者)可以将它拥有的页授权(grant)给接收者的地址空间。然后,授权者地址空间的这个已经授权的页会被移除。Grant 操作仅在非常特殊的情况下才使用:当很多页面要通过一个控制子系统映像到其他系统,而不想增加这个子系统地址空间的负担时,用到 grant 操作。

(2) Map:倘若接收者同意,地址空间的拥有者可以将自己的任何一页映像到其他地址空间。它和 grant 不同之处在于,map 后的页仍然保留在映射方这里。

(3) Flush:地址空间的所有者可以刷新任何属于它自己的页。

地址空间的概念将内存管理和页面调度程序留在了微内核之外。仅有 grant,map 和 flush 操作保留在内核里。Map 和 flush 用于在微内核上实现内存管理和页面调度程序。

4.2 内存管理

L4 将内存管理留在了微内核之外。如图 3 所示,每个线程有一个相应的页面调度程序。页面

调度程序负责管理一个线程的地址空间。一旦线程发生了缺页错误，内核获得这个信息，阻塞线程，同时，一个缺页 IPC 消息以线程的名义发送到页面调度程序。接着页面调度程序响应映像，最后，解除线程的锁，线程得以继续运行。

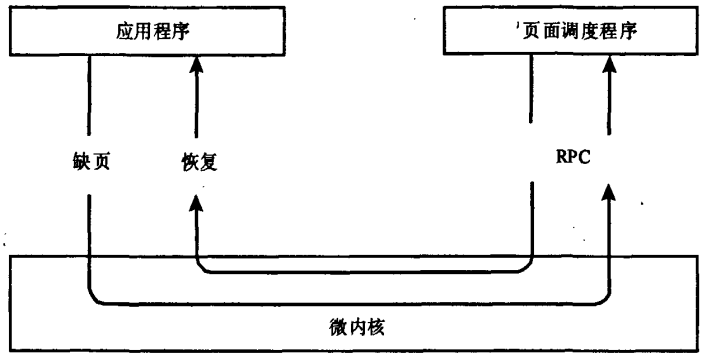

图 3 缺页处理

L4 还通过严格控制不必要的内存访问，从而减少 Cache 和 TLB miss 来提高进程间通信性能。因此新的 L4 版本还为每个线程静态的引入虚拟寄存器，虚拟寄存器为内核和用户态应用程序之间的信息交换提供了一种快速的手段。依赖处理器特性，这些寄存器映射到内存或是实际的处理器寄存器，提供在所有体系结构中最佳的硬件寄存器的使用。

4.3 分离的 API 和 ABI

加强了应用程序接口(API)和应用程序二进制接口(ABI)的独立，来支持可移植性，应用程序接口和一系列高级语言绑定，这样内核在所有的体系结构上都可以通用。

4.4 高效的进程间通信(IPC)技术

L4 的 IPC(Inter-Process Communication)用于通过值或引用来传递数值。它也用于同步，缺页处理，中断处理和异常处理。由于大部分的系统功能是在用户级服务器上实现的，因此高效的进程间通信对于 L4 保持优良的性能具有重要意义。L4 提供与系统调用性能相当的高效进程间通信机制，它可以完全执行在用户态，避免了两个不必要的特权级改变时的开销，消息无需占用核心存储空间。除高效的 IPC 机制外，L4 还在客户和服务器之间使用共享空间进行通信，进一步减少了通信开销。这些高效的进程间通信技术使得把核心功能移到服务器的策略非常实用且不影响性能。

4.5 内核接口页(KIP)

这里引入了一个由内核支持的页：内核接口页(Kernel Interface Page)，它包含所有系统调用和经常访问到的系统信息的函数入口。有了这个 KIP，系统调用就变得很简单了。如果下一代处理器有新的结构上的特性，那么通过替换这个页就可以直接利用这些特性了，使原来的编码可以自动的使用这些最新的处理器特性。

4.6 中断

L4 微内核把硬件中断处理成 IPC 消息。微内核把硬件当作是一些能够发送 IPC 消息给相关处理代码的线程，而把中断服务程序当作是一些正在接收这些 IPC 消息的线程。当一个硬件中断发生时，微内核会为这个中断产生一条消息并把此消息发送到和此中断相关联的用户进程中，然后由用户进程中的负责接收这条 IPC 消息的线程来处理这个硬件中断。这样，内核只负责产生中断消息，而不用涉及到具体的中断处理，从而使得中断处理的具体策略和内核隔离开来。

4.7 调度

L4 实现硬件优先，round-robin 的内部调度程序。线程调度由两个参数控制：时间片长度和线

程优先级。

在 L4 中时间片长度可以由每个线程自己制定。线程优先级决定了拥有某个特定优先级的线程属于哪个 round-robin 队列，修改一个线程的优先级就会将这个线程移到它应该在的那个 round-robin 队列。在某个特定的 round-robin 队列的线程，仅在没有更高优先级的线程可以运行时才被调度到。

5 结束语

微内核正以其移植性强，内核简洁，可扩展性好，支持最新硬件技术等独特魅力，吸引着越来越多的研究人员及机构参与到这项研究中来。它会给我们操作系统的研究带来新的契机。

参考文献：

[1] Gernot Heiser, Jerry Vocheteloo, L4 on Uni-and Multiprocessor Alpha

[2] Jochen Liedtke, Improving IPC by Kernel Design

[3] Jochen Liedtke, On μ-Kernel Construction

[4] Hermann Hartig, Michael Hohmuth, Jochen Liedtke, Sebastian Schonberg, Jean Wolter, The Performance of μ-Kernel-Based Systems

[5] Andreas Haeberlen1, Kevin Elphinstone, User-level Management of Kernel Memory

[6] Bernhard Kauer, Marcus Volp, Technische Universitat Dresden 01062 Dresden, Germany, {kauer, voelp}@os. inf. tu-dresden. de, Version: 0. 2, L4. Sec Preliminary Microkernel Reference Manual

L4微内核技术浅析

作者：陈斐
作者单位：江南计算技术研究所,江苏无锡,214083